Polynomial prenexing of QBFs with non-monotone boolean operators

Abdallah Saffidine, Andreas Herzig

Polynomial Prenexing of QBFs with Non-Monotone Boolean Operators

Abdallah Saffidine
Sydney, Australia

Andreas Herzig
IRIT, CNRS, France

Abstract

It is well-known that every quantified boolean formula (QBF) can be transformed into a prenex QBF whose only boolean operators are negation, conjunction, and disjunction. It is also well-known that the transformation is polynomial if the boolean operators of the original QBF are restricted to negation, conjunction, and disjunction. In contrast, up to now no polynomial transformation has been found when the original QBF contains other boolean operators such as biconditionals or exclusive disjunction. We define such a transformation and show that it is polynomial and preserves quantifier depth.

2012 ACM Subject Classification Theory of computation → Complexity theory and logic; Theory of computation → Problems, reductions and completeness

Keywords and phrases Quantified boolean formulas, QBF, prenexing, complexity, polynomial hierarchy

Acknowledgements We acknowledge discussions with several colleagues about the problem of prenexing with equivalences; in particular, an email exchange with Mikoláš Janota encouraged us to pursue our attempt.

1 Introduction

Provers for Quantified Boolean Formulas (QBF) require input formulas to be in prenex form: sequences of quantifiers over propositional variables (the prenex) followed by a boolean formula in conjunctive normal form CNF (the matrix). Any QBF whose boolean operators are restricted to negation, disjunction, and conjunction can be put in that form: one can move quantifiers outwards across these operators, replacing e.g. subformulas $\chi \wedge \exists x\varphi$ by $\exists x(\chi \wedge \varphi)$. The resulting formula is logically equivalent to the original formula if x does not occur free in χ, which can always be ensured by renaming. By iterating the application of these equivalences a formula in prenex form is obtained whose length is linear in the length of the original formula.

Moving quantifiers outwards is less obvious when formulas contain other boolean operators. The reason is that when a formula only contains negation, disjunction, and conjunction then one can associate a unique polarity to each subformula; this fails to hold when the formula contains operators that are non-monotone, such as the biconditional $\leftrightarrow$ and the exclusive disjunction $\oplus$. One can still move quantifiers outwards by eliminating the operators in a preprocessing step. For example, $\chi \leftrightarrow \exists x\varphi$ is first rewritten to $(\chi \wedge \exists x\varphi) \vee (\neg\chi \wedge \forall x\neg\varphi)$; then x is renamed to x^- in the second conjunct $\forall x\neg\varphi$, resulting in $(\chi \wedge \exists x\varphi) \vee (\neg\chi \wedge \forall x^-\neg\varphi[\{x/x^-\}])$; finally both quantifiers are prenexed. This, however, leads to exponential growth when non-monotone operators are nested.

One solution is to implement solvers accepting formulas that are not in prenex form; see e.g. [3, 4, 6, 8, 14]. We here take another route and define a polynomial transformation into prenex form. In the past several such transformations were introduced and studied in the literature [1, 2, 9, 12]. They improve over the naive preprocessing-based algorithm we have sketched above. Nevertheless, when formulas contain nested quantifiers and biconditionals then the resulting prenex formulas are exponentially bigger in the worst case. To the best of our knowledge, no polynomial prenexing transformation is known when the language contains biconditionals.

In this paper we show how one can put QBFs containing biconditionals—and more generally n-ary boolean operators—in prenex form. Our transformation draws inspiration from Tseytin's

transformation for propositional logic [15] and its generalisation to first-order predicate logic by Plaisted and Greenbaum [11]. (More recently, another polynomial transformation has been proposed by Harwath [5].) Contrarily to what one may expect, it is not obvious to adapt the Plaisted-Greenbaum transformation from predicate logic formulas to QBFs. Indeed, it makes use of fresh predicates in order to abbreviate subformulas, and it is essential that these predicates have object variables as arguments: there is no counterpart for such a device in the language of QBFs. Let us illustrate this by means of the formula $\exists z((\exists x((x \leftrightarrow z) \wedge \neg x)) \leftrightarrow (\exists y((y \leftrightarrow z) \wedge y)))$. A naive adaptation of the Plaisted-Greenbaum transformation would result in $\exists zxypq((p \leftrightarrow ((x \leftrightarrow z) \wedge \neg x)) \wedge (q \leftrightarrow ((y \leftrightarrow z) \leftrightarrow y)) \wedge (p \leftrightarrow q))$, with fresh variables p and q. However, the latter fails to be satisfiability-equivalent to the former.

Let us illustrate our transformation by the formula

$$\chi \oplus \exists x \varphi$$

where x does not occur in χ. We first apply Shannon-expansion, that is, we rewrite the formula so that we distinguish the two possible truth values that $\exists x \varphi$ can take:

$$(\exists x \varphi \wedge (\chi \oplus \top)) \vee (\forall x \neg \varphi \wedge (\chi \oplus \bot)).$$

We then rename one of the variables in the two quantifications over x by introducing fresh variables x^+ and x^-, resulting in

$$(\exists x^+ \exists x (\ \varphi \wedge (x \leftrightarrow x^+) \wedge (\chi \oplus \top))) \vee\ (\forall x^- \exists x (\neg \varphi \wedge (x \leftrightarrow x^-) \wedge (\chi \oplus \bot)))\,.$$

The latter can then be prenexed to

$$\forall x^- \exists x^+ \exists x((\varphi \wedge (x \leftrightarrow x^+) \wedge (\chi \oplus \top)) \vee (\neg \varphi \wedge (x \leftrightarrow x^-) \wedge (\chi \oplus \bot))),$$

which is propositionally equivalent to

$$\forall x^- \exists x^+ \exists x\ ((\varphi \rightarrow (x \leftrightarrow x^+)) \wedge (\neg \varphi \rightarrow (x \leftrightarrow x^-)) \wedge (\chi \oplus \varphi))\,.$$

Finally, we abbreviate φ by a fresh p, resulting in

$$\forall x^- \exists x^+ \exists x \exists p\, ((p \leftrightarrow \varphi) \wedge (p \rightarrow (x \leftrightarrow x^+)) \wedge (\neg p \rightarrow (x \leftrightarrow x^-)) \wedge\ (\chi \oplus p)).$$

The last formula is logically equivalent to the initial formula $\chi \oplus \exists x \varphi$ under the proviso that x does not occur free in χ.

In this paper we generalise the above transformation to quantified boolean formulas where quantifiers and n-ary boolean operators are arbitrarily nested. We call the language of such formulas *full QBF*. We define full QBFs and their semantics in Section 2. We then introduce our transformation: we first prenex outermost quantifiers (Section 3) and then all of them (Section 4). In Section 5 we discuss implementation and evaluation of our transformation and Section 6 concludes.

2 Full QBFs

Our language of full QBFs has *quantifier blocks*: quantifiers applying to finite sets of propositional variables. These quantifiers can occur in the scope of boolean operators. The latter are arbitrary: we suppose given some set of boolean functions f^k, each of some arity $k \geq 0$, whose syntactical counterpart are boolean operators noted $\mathbf{f}^k$. We suppose that the set of these operators contains the zero-ary operator $\bot$, the unary $\neg$, and the binary $\wedge$, $\rightarrow$, and $\leftrightarrow$.

2.1 Language

Given a countably infinite set of propositional variables Var, the language $\mathcal{L}$ of full QBFs is formally defined to be the smallest set such that:

- $p \in \mathcal{L}$ for every $p \in \mathsf{Var}$;
- $\mathbf{f}^k(\varphi_1, \ldots, \varphi_k) \in \mathcal{L}$ if $\varphi_1, \ldots, \varphi_k \in \mathcal{L}$ and $\mathbf{f}^k$ is a k-ary boolean operator;
- $\exists X\varphi \in \mathcal{L}$ if $\varphi \in \mathcal{L}$ and X is a finite subset of Var;
- $\forall X\varphi \in \mathcal{L}$ if $\varphi \in \mathcal{L}$ and X is a finite subset of Var.

We say that $\exists$ is the dual of $\forall$, and vice versa. Note that in $\forall X\varphi$ and $\exists X\varphi$ the set X can be empty. We write $\exists x\varphi$ instead of $\exists\{x\}\varphi$ and $\forall x\varphi$ instead of $\forall\{x\}\varphi$. The abbreviation $\bigwedge_{k\leq n} \varphi_k$ stands for the conjunction $\varphi_1 \wedge \cdots \wedge \varphi_n$, where we suppose that it equals $\top$ when n is zero.

The set of boolean formulas $\mathcal{L}_{\mathsf{bool}}$ is the set of full QBFs from $\mathcal{L}$ without $\exists$ and $\forall$. We use $\beta, \beta', \ldots$ to denote boolean formulas.

2.2 Bound Variables, Free Variables, and Substitutions

The *set of propositional variables occurring in* φ is noted $\mathsf{Var}(\varphi)$; its set of *free variables* is $\mathsf{Free}(\varphi)$ and its set of *bounded variables* is $\mathsf{Bound}(\varphi)$. The *number of blocks* in φ, noted $\mathsf{NBlock}(\varphi)$, is the number of occurrences of quantified subformulas $QX\psi$ in φ. The *number of quantified variables* in φ, noted $\mathsf{NQVar}(\varphi)$, is the sum of the lengths of the occurrences of blocks in φ. For example, for the formula $\varphi = \exists x(\psi \wedge \neg\exists\{x, y\}\chi) \wedge \neg\forall y\rho$ we have $\mathsf{NBlock}(\varphi) = 3$ and $\mathsf{NQVar}(\varphi) = 4$.

A *substitution* is a set of couples $\sigma = \{p_1/\varphi_1, \ldots, p_k/\varphi_k\}$ such that $p_i \neq p_j$ for $i \neq j$. Its application to a formula φ, noted $\varphi[\sigma]$, simultaneously replaces all free occurrences of the variables p_i in φ by φ_i. Instead of $\varphi[\{p_1/\varphi_1, \ldots, p_k/\varphi_k\}]$ we sometimes write $\varphi[p_1/\varphi_1, \ldots, p_k/\varphi_k]$ or $\varphi[p_i/\varphi_i \ : \ i \leq k]$. We suppose that substitutions have narrow scope. For example, $\varphi \wedge \psi[\sigma]$ stands for $\varphi \wedge (\psi[\sigma])$.

2.3 Metrics

The *length* of a formula $\varphi \in \mathcal{L}$, noted $\|\varphi\|$, is the number of logical and non-logical symbols occurring in φ, where propositional variables and logical operators both have length 1 and negations and parentheses do not count. Formally, the inductive definition is: $\|p\| = 1$; $\|\mathbf{f}^n(\varphi_1, \ldots, \varphi_n)\| = 1 + \sum_{i\leq k} \|\varphi_i\|$; and $\|\exists X\varphi\| = \|\forall X\varphi\| = 1 + |X| + \|\varphi\|$. For example, the length of $p \leftrightarrow q$ is 3 and the length of $\exists\{x, y\}\ (x \rightarrow p)$ is $1 + |\{x, y\}| + \|x \rightarrow p\| = 1 + 2 + 3 = 6$.

Quantifier depth of φ, noted $\mu(\varphi)$, is defined by:

$$\begin{aligned} \mu(p) &= 0; \\ \mu(\mathbf{f}^n(\varphi_1, \ldots, \varphi_n)) &= \max(\mu(\varphi_1), \ldots, \mu(\varphi_n)); \\ \mu(\exists X\varphi) = \mu(\forall X\varphi) &= 1 + \mu(\varphi); \end{aligned}$$

where we suppose that $\max(\mu(\varphi_1), \ldots, \mu(\varphi_n))$ equals 0 when n equals 0. Hence the quantifier depth of a formula is zero if and only if it is boolean.

Our definition of quantifier depth differs from the more standard notion of alternation depth: there are formulas with $\mu(\varphi) > 1$ where $\exists$ and $\forall$ do not alternate, such as $\varphi = \exists X\exists Y\psi$. The reason for our choice is that quantifier depth is a more natural notion when there are non-monotone boolean operators. Indeed, when there are only monotone boolean operators then it makes sense to say that the depth of e.g. $\exists x\neg\exists y x \vee y$ should be 2 because to each subformula one can associate a unique polarity. However, it is more difficult to associate a meaningful notion of an alternation depth to formulas such as $\exists x((\exists y y) \leftrightarrow x)$.

2.4 Prenex Normal Form

A *prefix* is a sequence of $\exists X$ and $\forall Y$. A QBF is in *prenex form* if it is of the form $\Lambda\beta$ where Λ is a prefix and $\beta \in \mathcal{L}_{\mathsf{bool}}$. Hence the quantifier depth of a formula in prenex form is the number of occurrences of $\exists$ and $\forall$ in the prefix.

2.5 Semantics

A *valuation* is a mapping from Var to $\{0,1\}$. We use $V, W, \ldots$ for valuations.

We say that two valuations V and V' *agree* on a set of propositional variables $X \subseteq \mathsf{Var}$ if their restrictions to X are identical, that is, if $V|_X = V'|_X$.

Valuations are extended from Var to complex formulas of $\mathcal{L}$ as follows:

$$V\left(\mathbf{f}^k(\varphi_1,\ldots,\varphi_k)\right) = f^k(V(\varphi_1),\ldots,V(\varphi_k));$$

$$V(\exists X\varphi) = \begin{cases} 1 & \text{if } V'(\varphi) = 1 \text{ for some } V' \text{ agreeing with } V \text{ on } \mathsf{Var}\setminus X,\\ 0 & \text{otherwise;}\end{cases}$$

$$V(\forall X\varphi) = \begin{cases} 1 & \text{if } V'(\varphi) = 1 \text{ for every } V' \text{ agreeing with } V \text{ on } \mathsf{Var}\setminus X,\\ 0 & \text{otherwise.}\end{cases}$$

A formula $\varphi \in \mathcal{L}$ is *satisfiable* if $V(\varphi) = 1$ for some valuation V; it is *valid* if $V(\varphi) = 1$ for every V. Two formulas φ and ψ are *logically equivalent*, written $\varphi \equiv \psi$, if $\varphi \leftrightarrow \psi$ is valid; that is, if $V(\varphi) = V(\psi)$ for every valuation V. They are *equisatisfiable* if φ is satisfiable exactly when ψ is satisfiable; and they are *equivalid* if φ is valid exactly when ψ is valid.

We state some logical equivalences that are going to be useful. Remember that a formula ψ is *free (to be substituted) for* p in φ if it is not the case that there is a free occurrence of p in φ that is in the scope of a quantifier in φ binding some free variable of ψ. A particular case is when no occurrence of p in φ is in the scope of a quantifier: then any ψ is free for p in φ.

▶ **Fact 1.** *Let φ be a full QBF in which ψ is free for p. Then:*

1. $\varphi[p/\psi] \equiv (\varphi[p/\top] \wedge \psi) \vee (\varphi[p/\bot] \wedge \neg\psi)$*;*
2. *If $p \notin \mathsf{Free}(\psi)$ then $\varphi[p/\psi] \equiv \exists p\,((p \leftrightarrow \psi) \wedge \varphi)$.*

Proof. For item 1, φ is logically equivalent to $(\varphi[p/\top] \wedge p) \vee (\varphi[p/\bot] \wedge \neg p)$. As ψ is free for p in both formulas, the substitution $\{p/\psi\}$ preserves this: $\varphi[p/\psi]$ is logically equivalent to $((\varphi[p/\top] \wedge p) \vee (\varphi[p/\bot] \wedge \neg p))[p/\psi]$. The latter is logically equivalent to $(\varphi[p/\top] \wedge \psi) \vee (\varphi[p/\bot] \wedge \neg\psi)$.

For item 2, according to the truth condition for $\exists$ the right hand side $\exists p\,((p \leftrightarrow \psi) \wedge \varphi)$ is logically equivalent to the disjunction $(\varphi \wedge (p \leftrightarrow \psi))\,[p/\top] \vee (\varphi \wedge (p \leftrightarrow \psi))\,[p/\bot]$. The latter equals $(\varphi[p/\top] \wedge (\top \leftrightarrow \psi)) \vee (\varphi[p/\bot] \wedge (\bot \leftrightarrow \psi))$ because p does not occur free in ψ; which simplifies to $(\varphi[p/\top] \wedge \psi) \vee (\varphi[p/\bot] \wedge \neg\psi)$. This is logically equivalent to the left hand side $\varphi[p/\psi]$ by item 1. ■

▶ **Fact 2.** *Let $\sigma = \{p_1/\varphi_1, \ldots, p_k/\varphi_k\}$ be a substitution. Let $p \in \mathsf{Var}$ such that for every $i \leq k$, $p \notin \{p_i\} \cup \mathsf{Free}(\varphi_i)$. Then $\varphi[\sigma \cup \{p/\psi\}] \equiv \varphi[\sigma][p/\psi]$.*

▶ **Fact 3.** *Let $\forall X\varphi$ be a full QBF. Let $X^- = \{x^- \;:\; x \in X\}$ be fresh. Then*

$$\forall X\varphi \equiv \forall X^-\exists X\left(\varphi \wedge \bigwedge_{x\in X}(x \leftrightarrow x^-)\right).$$

2.6 Decision Problems

The satisfiability problem is to decide whether a given formula $\varphi \in \mathcal{L}$ is satisfiable; and the validity problem is to decide whether φ is valid. The model checking problem is to decide, given a valuation V and a formula $\varphi \in \mathcal{L}$, whether $V(\varphi) = 1$.

It is known that for QBFs in prenex form, the satisfiability problem for formulas of quantifier depth k is $\Sigma^{\mathsf{P}}_{k+1}$-complete and the validity problem is Π^{P}_{k+1}-complete. Our transformation will establish that the same is the case for full QBFs. It will moreover establish that the model checking problem for full QBFs of quantifier depth k is Θ^{P}_{k}-complete. This contrasts with prenex QBFs, for which model checking is not Θ^{P}_{k}-hard under standard computational complexity assumptions.

3 Prenexing Outermost Quantifiers

An outermost quantifier is not in the scope of any other quantifier. The first step of our transformation is to prenex all such quantifiers without doubling the occurrences of the quantified subformula. This requires some fresh auxiliary variables.

We start by a lemma that moves one quantifier outwards and fuses it with an existing quantifier $\exists Y$ (for which Y may be empty).

▶ **Lemma 1.** *Let $\psi, \varphi \in \mathcal{L}$ be full QBFs. Let $p \in \mathsf{Var}(\psi) \setminus \mathsf{Free}(\varphi)$ be not in the scope of any quantifier in ψ. Let $Y \subseteq \mathsf{Var}(\psi) \setminus \mathsf{Var}(\varphi)$. Let $X^- = \{x^- \ : \ x \in X\}$ and $X^+ = \{x^+ \ : \ x \in X\}$ be sets of fresh variables. Let $\sigma = \{x/x^+ \ : \ x \in X\}$. Then*

$$\exists Y\,(\psi[p/\exists X\varphi]) \equiv \forall X^-\ \exists Y \cup X^+ \cup \{p\} \left(\psi \wedge (p \leftrightarrow \varphi[\sigma]) \wedge \left(\bigwedge_{x \in X} (\neg p \rightarrow (x^+ \leftrightarrow x^-)) \right) \right);$$

$$\exists Y\,(\psi[p/\forall X\varphi]) \equiv \forall X^-\ \exists Y \cup X^+ \cup \{p\} \left(\psi \wedge (p \leftrightarrow \varphi[\sigma]) \wedge \left(\bigwedge_{x \in X} (\ \ p \rightarrow (x^+ \leftrightarrow x^-)) \right) \right).$$

Proof. For the first equivalence we give a sequence of formulas whose first is the left side and whose last is the right side and which are all logically equivalent:

1. $\exists Y\ (\psi[p/\exists X\varphi])$
2. $\exists Y\ ((\psi[p/\top] \wedge \exists X\varphi) \vee (\psi[p/\bot] \wedge \neg\exists X\,\varphi))$
   by Fact 1.1 (applies: p is not in the scope of any quantifier in ψ, so $\exists X\varphi$ is free for p in ψ)
3. $(\exists Y\,(\psi[p/\top] \wedge \exists X\varphi)) \vee \exists Y\ (\psi[p/\bot] \wedge \forall X\neg\varphi)$ distribution of $\exists Y$ over $\vee$
4. $(\exists Y(\psi[p/\top] \wedge \exists X\varphi)) \vee ((\exists Y\psi[p/\bot]) \wedge \forall X\neg\varphi)$ distribution of $\exists Y$ over $\wedge$ (correct because $Y \cap \mathsf{Free}(\varphi) = \emptyset$)
5. $(\exists Y(\psi[p/\top] \wedge \exists X\varphi)) \vee ((\exists Y\psi[p/\bot]) \wedge \forall X^-\ \exists X\,(\neg\varphi \wedge \bigwedge_{x\in X}(x \leftrightarrow x^-)))$ by Fact 3, for fresh X^-
6. $(\exists Y(\psi[p/\top] \wedge \exists X^+\varphi[\sigma])) \vee ((\exists Y\psi[p/\bot]) \wedge \forall X^-\ \exists X^+\,(\neg\varphi[\sigma] \wedge \bigwedge_{x\in X}(x^+ \leftrightarrow x^-)))$
   by definition of σ
7. $\forall X^-\ \left((\exists Y(\psi[p/\top] \wedge \exists X^+\varphi[\sigma])) \vee \left((\exists Y\psi[p/\bot]) \wedge \exists X^+ \left(\neg\varphi[\sigma] \wedge \bigwedge_{x\in X}(x^+ \leftrightarrow x^-) \right) \right) \right)$
   distribution of $\forall X^-$ over $\wedge$ and $\vee$ (correct because X^- are fresh)
8. $\forall X^-\ \left((\exists Y(\psi[p/\top] \wedge \exists X^+\varphi[\sigma])) \vee \exists Y \left(\psi[p/\bot] \wedge \exists X^+ \left(\neg\varphi[\sigma] \wedge \bigwedge_{x\in X}(x^+ \leftrightarrow x^-) \right) \right) \right)$
   distribution of $\exists Y$ over $\wedge$ (correct because $Y \cap \mathsf{Free}(\varphi[\sigma]) = \emptyset$)

9. $\forall X^- \left((\exists Y \cup X^+ (\psi[p/\top] \wedge \varphi[\sigma])) \vee \exists Y \cup X^+ \left(\psi[p/\bot] \wedge \neg\varphi[\sigma] \wedge \bigwedge_{x \in X} (x^+ \leftrightarrow x^-) \right) \right)$

distribution of $\exists X^+$ over $\wedge$ (correct because $X^+ \cap \mathsf{Free}(\psi) = \emptyset$)

10. $\forall X^- \exists Y \cup X^+ \left((\psi[p/\top] \wedge \varphi[\sigma]) \vee \left(\psi[p/\bot] \wedge \neg\varphi[\sigma] \wedge \bigwedge_{x \in X} (x^+ \leftrightarrow x^-) \right) \right)$

distribution of $\exists Y \cup X^+$ over $\vee$

11. $\forall X^- \exists Y \cup X^+ \left(((\psi[p/\top] \wedge \varphi[\sigma]) \vee (\psi[p/\bot] \wedge \neg\varphi[\sigma])) \wedge \left(\neg\varphi[\sigma] \rightarrow \bigwedge_{x \in X} (x^+ \leftrightarrow x^-) \right) \right)$

propositional calculus

12. $\forall X^- \exists Y \cup X^+ \ ((\psi[p/\varphi[\sigma]] \wedge (\neg\varphi[\sigma] \rightarrow \bigwedge_{x \in X} (x^+ \leftrightarrow x^-))))$

by Fact 1.1 (applies: p is not in the scope of any quantifier in ψ, so $\varphi[\sigma]$ is free for p in ψ)

13. $\forall X^- \exists Y \cup X^+ \ ((\psi \wedge \bigwedge_{x \in X} (\neg p \rightarrow (x^+ \leftrightarrow x^-)))\,[p/\varphi[\sigma]])$

substitution (correct: $X^+ \cap \mathsf{Free}(\psi) = \emptyset$ and $p \notin X^-$)

14. $\forall X^- \exists Y \cup X^+ \cup \{p\} \left(\psi \wedge (p \leftrightarrow \varphi[\sigma]) \wedge \bigwedge_{x \in X} (\neg p \rightarrow (x^+ \leftrightarrow x^-)) \right)$

by Fact 1.2 (applies: p not in the scope of any quantifier in ψ and $p \notin \mathsf{Free}(\varphi[\sigma])$)

For the second equivalence we use that $\exists Y \psi[p/\forall X \varphi]$ is logically equivalent to $\exists Y \, ((\psi[p/\neg q])[q/\exists X \neg\varphi])$ if q is fresh. Then the first case applies: the latter is logically equivalent to

$$\forall X^- \ \exists Y \cup X^+ \cup \{q\} \left(\psi[p/\neg q] \wedge (q \leftrightarrow \neg\varphi[\sigma]) \wedge \left(\bigwedge_{x \in X} (\neg q \rightarrow (x^+ \leftrightarrow x^-)) \right) \right),$$

which is logically equivalent to

$$\forall X^- \ \exists Y \cup X^+ \cup \{p\} \left(\psi \wedge (p \leftrightarrow \varphi[\sigma]) \wedge \left(\bigwedge_{x \in X} (\ \ p \rightarrow (x^+ \leftrightarrow x^-)) \right) \right).$$

■

We are going to iterate the application of Lemma 1 in order to prenex all outermost quantifiers. First of all, we observe that we can identify all outermost quantifiers of a given full QBF φ.

▶ **Fact 4.** *For every full QBF $\varphi \in \mathcal{L}$ there exist a boolean formula $\beta \in \mathcal{L}_{\mathsf{bool}}$ and a substitution $\{p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k\}$ such that*

$$\varphi = \beta[p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k],$$

$\|\varphi\| = \|\beta\| + \sum_{i \leq k} |X_i| + \sum_{i \leq k} \|\varphi_i\|$, and $\mu(\varphi) = \max_{i \leq k}(1 + \mu(\varphi_i))$. The formula β can be chosen such that $p_i \notin \bigcup_{j \leq k} \mathsf{Free}(\varphi_j)$.

The above fact also holds when φ is boolean: then $k = 0$ (the substitution is empty), φ equals β, and $\mu(\varphi) = 0$.

We are ready to define a transformation simultaneously prenexing all outermost quantifiers.

▶ **Definition 2.** *Let $\varphi \in \mathcal{L}$ be a full QBF. Let $\beta \in \mathcal{L}_{\mathsf{bool}}$ and $\{p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k\}$ be as in Fact 4. For $i \leq k$ and $x \in X_i$, let x_i^+ and x_i^- be fresh variables. Let $\sigma_i = \{x/x_i^+ \ : \ x \in X_i\}$.*

$$\mathtt{t}_\vee(\varphi) = \left(\left(\bigwedge_{i \leq k} (p_i \leftrightarrow \varphi_i[\sigma_i]) \right) \wedge \bigwedge_{i \leq k, Q_i = \exists} \left(\neg p_i \rightarrow \bigwedge_{x \in X_i} (x_i^+ \leftrightarrow x_i^-) \right) \wedge \bigwedge_{i \leq k, Q_i = \forall} \left(\ \ p_i \rightarrow \bigwedge_{x \in X_i} (x_i^+ \leftrightarrow x_i^-) \right) \right) \wedge \beta;$$

$$\mathtt{t}_\exists(\varphi) = \left(\left(\bigwedge_{i\le k}(p_i \leftrightarrow \varphi_i[\sigma_i])\right) \wedge \bigwedge_{i\le k, Q_i=\exists}\left(\neg p_i \rightarrow \bigwedge_{x\in X_i}(x_i^+ \leftrightarrow x_i^-)\right) \wedge \bigwedge_{i\le k, Q_i=\forall}\left(\ p_i \rightarrow \bigwedge_{x\in X_i}(x_i^+ \leftrightarrow x_i^-)\right)\right) \rightarrow \beta;$$

$$\mathsf{N}(\varphi) = \{x_i^- \ :\ x \in X_i, i \le k\};$$

$$\mathsf{P}(\varphi) = \{x_i^+ \ :\ x \in X_i, i \le k\} \cup \{p_i \ :\ i \le k\}.$$

The substitutions σ_i replace the elements of X_i by fresh variables and thereby ensure that in $\mathtt{t}_\forall(\varphi)$ and $\mathtt{t}_\exists(\varphi)$, none of the variables of X_i is quantified.

▶ **Example 3.** Let

$$\begin{aligned}\varphi &= \exists x(\psi \wedge \neg\exists x\chi) \wedge \neg\forall y\rho\\ &= (p_1 \wedge \neg p_2)[p_1/\exists x(\psi \wedge \neg\forall x\chi), p_2/\forall y\rho],\end{aligned}$$

where ψ, χ, ρ are all boolean. Then $\mathsf{N}(\varphi) = \{x^-, y^-\}$, $\mathsf{P}(\varphi) = \{x^+, y^+, p_1, p_2\}$, and

$$\mathtt{t}_\exists(\varphi) = \begin{pmatrix} (p_1 \leftrightarrow (\psi \wedge \neg\exists x\chi)[x/x^+]) \\ \wedge\ (\neg p_1 \rightarrow (x^+ \leftrightarrow x^-)) \wedge \top \\ \wedge\ (p_2 \leftrightarrow \rho[y/y^+]) \\ \wedge\ \top \wedge (p_2 \rightarrow (y^+ \leftrightarrow y^-)) \end{pmatrix} \rightarrow (p_1 \wedge \neg p_2).$$

The formula

$$\exists\mathsf{N}(\varphi)\forall\mathsf{P}(\varphi)\mathtt{t}_\exists(\varphi) = \exists\{x^-, y^-\}\forall\{x^+, y^+, p_1, p_2\}\mathtt{t}_\exists(\varphi)$$

is logically equivalent to φ, as will follow from the next lemma.

▶ **Lemma 4.** *Let $\varphi \in \mathcal{L}$ be a full QBF and let Q be either $\forall$ or $\exists$. Then:*

1. *$Q\mathsf{N}(\varphi)Q'\mathsf{P}(\varphi)\,\mathtt{t}_Q(\varphi) \equiv \varphi$, where Q' is the dual of Q.*
2. *If φ is boolean then $\mu(\mathtt{t}_Q(\varphi)) = 0$, else $\mu(\mathtt{t}_Q(\varphi)) = \mu(\varphi) - 1$.*
3. *$\|\mathtt{t}_\forall(\varphi)\| + 3\mathsf{NQVar}(\mathtt{t}_\forall(\varphi)) + 6\mathsf{NBlock}(\mathtt{t}_\forall(\varphi))$*
   $\le \|\varphi\| + 3\mathsf{NQVar}(\varphi) + 6\mathsf{NBlock}(\varphi)$.

Proof. Let $\varphi \in \mathcal{L}$ be a full QBF. Let $\beta \in \mathcal{L}_{\mathsf{bool}}$ and $\{p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k\}$ be as in Definition 2 (and Fact 4).

Let us prove *item (1)* for the case where Q is $\forall$. We define for all $i \le k$:

$$\mathsf{N}_{\le i} = \{x_j^- \ :\ j \le i \text{ and } x \in X_j\},$$

$$\mathsf{P}_{\le i} = \{x_j^+ \ :\ j \le i \text{ and } x \in X_j\} \cup \{p_j \ :\ j \le i\}.$$

We show by induction on $k \ge 0$ that φ is logically equivalent to $\forall\mathsf{N}_{\le k}\,\exists\mathsf{P}_{\le k}\mathtt{t}_\forall(\varphi)$, that is, to

$$\forall\mathsf{N}_{\le k}\ \exists\mathsf{P}_{\le k}\begin{pmatrix} \beta \wedge \left(\bigwedge_{i\le k}(p_i \leftrightarrow \varphi_i[\sigma_i])\right) \\ \wedge \bigwedge_{\substack{i\le k\\ Q_i=\exists}}\left(\neg p_i \rightarrow \bigwedge_{x\in X_i}(x_i^+ \leftrightarrow x_i^-)\right) \\ \wedge \bigwedge_{\substack{i\le k\\ Q_i=\forall}}\left(\ p_i \rightarrow \bigwedge_{x\in X_i}(x_i^+ \leftrightarrow x_i^-)\right) \end{pmatrix}.$$

In the base case $k = 0$ the full QBF φ is boolean. Hence the set of p_i, the set of x_i^+, and the set of x_i^- are all empty and φ and $\mathtt{t}_\forall(\varphi)$ both equal β; therefore the equivalence trivially holds. For the induction step we give the following sequence of logically equivalent formulas:

1. φ
2. $\beta[p_i/Q_iX_i\varphi_i \ : \ i \leq k+1]$ by Fact 4
3. $(\beta[p_i/Q_iX_i\varphi_i \ : \ i \leq k])\,[p_{k+1}/Q_{k+1}X_{k+1}\varphi_{k+1}]$ by Fact 2 (applies: p_{k+1} is distinct from $p_1, \dots p_k$ and $p_{k+1} \notin \bigcup_{i\leq k} \mathsf{Free}(\varphi_i)$)
4. $$\left(\forall \mathsf{N}_{\leq k} \exists \mathsf{P}_{\leq k} \left(\begin{array}{r} \beta \wedge \left(\bigwedge_{i\leq k} (p_i \leftrightarrow \varphi_i[\sigma_i])\right) \\ \wedge \bigwedge_{\substack{i\leq k \\ Q_i=\exists}} \left(\neg p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-)\right) \\ \wedge \bigwedge_{\substack{i\leq k \\ Q_i=\forall}} \left(\ p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-)\right) \end{array}\right)\right)[p_{k+1}/Q_{k+1}X_{k+1}\varphi_{k+1}]$$ by induction hypothesis
5. $$\forall \mathsf{N}_{\leq k} \exists \mathsf{P}_{\leq k} \left(\left(\begin{array}{r} \beta \wedge \left(\bigwedge_{i\leq k} (p_i \leftrightarrow \varphi_i[\sigma_i])\right) \\ \wedge \bigwedge_{\substack{i\leq k \\ Q_i=\exists}} \left(\neg p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-)\right) \\ \wedge \bigwedge_{\substack{i\leq k \\ Q_i=\forall}} \left(\ p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-)\right) \end{array}\right)[p_{k+1}/Q_{k+1}X_{k+1}\varphi_{k+1}]\right)$$ because $\mathsf{N}_{\leq k} \cup \mathsf{P}_{\leq k}$ and $\{p_{k+1}\} \cup \mathsf{Free}(\varphi_{k+1})$ are disjoint ($\mathsf{N}_{\leq k} \cup \mathsf{P}_{\leq k}$ being fresh)
6. $$\forall \mathsf{N}_{\leq k} \forall X_{k+1}^- \exists \mathsf{P}_{\leq k+1} \cup X_{k+1}^+ \cup \{p_{k+1}\} \left(\begin{array}{r} \beta \wedge \left(\bigwedge_{i\leq k+1} (p_i \leftrightarrow \varphi_i[\sigma_i])\right) \\ \wedge \bigwedge_{\substack{i\leq k+1 \\ Q_i=\exists}} \left(\neg p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-)\right) \\ \wedge \bigwedge_{\substack{i\leq k+1 \\ Q_i=\forall}} \left(\ p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-)\right) \end{array}\right)$$ by Lemma 1 (applies: the sets $\mathsf{P}_{\leq k}$ and $\mathsf{Free}(\varphi_{k+1})$ are disjoint)
7. $\forall \mathsf{N}_{\leq k+1} \exists \mathsf{P}_{\leq k+1} \mathtt{t}_\forall(\varphi)$.

The case where Q is $\exists$ follows directly from the case where Q is $\forall$: it suffices to show $\mathtt{t}_\exists(\varphi) \equiv \neg\mathtt{t}_\forall(\neg\varphi)$, that is, that $\mathtt{t}_\exists(\beta[p_1/\exists X_1\varphi_1, \dots, p_k/\exists X_k\varphi_k]) \equiv \neg\mathtt{t}_\forall((\neg\beta)[p_1/\exists X_1\varphi_1, \dots, p_k/\exists X_k\varphi_k])$.

Concerning *item 2* of the lemma about quantifier depth, if $k = 0$ then φ is boolean and $\mu(\varphi) = \mu(\mathtt{t}_\exists(\varphi)) = \mu(\mathtt{t}_\forall(\varphi)) = 0$; otherwise

$$\begin{aligned} \mu(\varphi) &= 1 + \max_{i\leq k} \mu(\varphi_i) \\ &= 1 + \mu(\mathtt{t}_\exists(\varphi)) \\ &= 1 + \mu(\mathtt{t}_\forall(\varphi)). \end{aligned}$$

Finally, concerning *item 3* of the lemma about formula length, let us prove that

$$\|\mathtt{t}_\forall(\varphi)\| \leq \|\varphi\| + 6(\mathsf{NBlock}(\varphi) - \mathsf{NBlock}(\mathtt{t}_\forall(\varphi))) + 3(\mathsf{NQVar}(\varphi) - \mathsf{NQVar}(\mathtt{t}_\forall(\varphi))).$$

We have:

$$
\begin{aligned}
&\|\mathtt{t}_\forall(\varphi)\| \\
&= \|\beta\| + 1 + \left\| \bigwedge_{i\le k} (p_i \leftrightarrow \varphi_i[\sigma_i]) \right\| + \left\| \begin{array}{l} \displaystyle\bigwedge_{i\le k, Q_i=\exists} \left(\neg p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-) \right) \\ \displaystyle\wedge \bigwedge_{i\le k, Q_i=\forall} \left(p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-) \right) \end{array} \right\| \\
&\le \|\beta\| + 1 + \left\| \bigwedge_{i\le k} (p_i \leftrightarrow \varphi_i[\sigma_i]) \right\| + \left\| \bigwedge_{i\le k} \left(\neg p_i \rightarrow \bigwedge_{x\in X_i} (x_i^+ \leftrightarrow x_i^-) \right) \right\| \\
&\le \|\beta\| + \left(\sum_{i\le k} (3 + \|\varphi_i\|) \right) + \left(\sum_{i\le k} (3 + 4|X_i|) \right) \\
&\le \|\beta\| + 6k + \left(\sum_{i\le k} \|\varphi_i\| \right) + 4 \left(\sum_{i\le k} |X_i| \right) \\
&\le 6k + 3 \left(\sum_{i\le k} |X_i| \right) + \|\beta\| + \left(\sum_{i\le k} |X_i| \right) + \left(\sum_{i\le k} \|\varphi_i\| \right) \\
&\le 6k + 3 \left(\sum_{i\le k} |X_i| \right) + \|\varphi\| \\
&\le 6(\mathsf{NBlock}(\varphi) - \mathsf{NBlock}(\mathtt{t}_\forall(\varphi))) + 3(\mathsf{NQVar}(\varphi) - \mathsf{NQVar}(\mathtt{t}_\forall(\varphi))) + \|\varphi\| .
\end{aligned}
$$

The same upper bound for the length of $\mathtt{t}_\exists(\varphi)$ can be shown in the same way.
This ends the proof of Lemma 4. ■

4 Prenexing All Quantifiers

Lemma 4 allows us to prenex all outermost quantifiers of a full QBF. In order to prenex all quantifiers we iterate the transformations $\mathtt{t}_\forall(\varphi)$ and $\mathtt{t}_\exists(\varphi)$ of Definition 2, where we take care to avoid quantifier depth increase. The latter is achieved by choosing the appropriate among the two equivalences of Lemma 1 and by fusing quantifier blocks.

▶ **Definition 5.** *Let Q be a quantifier and let Q' be its dual. Let $\varphi \in \mathcal{L}$ be a full QBF.*

$$
\mathbf{T}_Q(\varphi) = \begin{cases} \varphi & \text{if } \varphi \text{ is boolean;} \\ Q'\mathsf{P}(\varphi)\cup\mathsf{N}(\psi)\mathbf{T}_{Q'}(\psi) & \text{otherwise, for } \psi = \mathtt{t}_Q(\varphi). \end{cases}
$$

Observe that the transformation is well-defined because the quantifier depth decreases in the induction step by Lemma 4.2. The freshness of the variables x_i^+ substituting the x_i of the subformulas φ_i according to Definition 2 ensures that the translation behaves well.

▶ **Example 6.** Let

$$
\begin{aligned}
\varphi &= \exists x(\psi \wedge \neg\exists x\chi) \wedge \neg\forall y\rho \\
&= (p_1 \wedge \neg p_2)[p_1/\exists x(\psi \wedge \neg\exists x\chi), p_2/\forall y\rho],
\end{aligned}
$$

where ψ, χ, and ρ are all boolean. First:

$$\mathtt{t}_\exists(\varphi) = \begin{pmatrix} p_1 \leftrightarrow (\psi[x/x^+] \wedge \neg\exists x\chi) \\ \wedge\ \neg p_1 \rightarrow (x^+ \leftrightarrow x^-) \\ \wedge\ \ p_2 \leftrightarrow \rho[y/y^+] \\ \wedge\ p_2 \rightarrow (y^+ \leftrightarrow y^-) \end{pmatrix} \rightarrow (p_1 \wedge \neg p_2),$$

$$\mathsf{N}(\varphi) = \{x^-, y^-\},$$

$$\mathsf{P}(\varphi) = \{x^+, y^+, p_1, p_2\}.$$

Second:

$$\mathtt{t}_\forall(\mathtt{t}_\exists(\varphi)) = \begin{pmatrix} p_3 \leftrightarrow \chi[x/z^+] \\ \wedge\ \neg p_3 \rightarrow (z^+ \leftrightarrow z^-) \end{pmatrix} \wedge \left(\begin{pmatrix} p_1 \leftrightarrow (\psi[x/x^+] \wedge \neg p_3) \\ \wedge\ \neg p_1 \rightarrow (x^+ \leftrightarrow x^-) \\ \wedge\ \ p_2 \leftrightarrow \rho[y/y^+] \\ \wedge\ p_2 \rightarrow (y^+ \leftrightarrow y^-) \end{pmatrix} \rightarrow (p_1 \wedge \neg p_2) \right),$$

$$\mathsf{N}(\mathtt{t}_\exists(\varphi)) = \{z^-\},$$

$$\mathsf{P}(\mathtt{t}_\exists(\varphi)) = \{z^+, p_3\}.$$

Finally, $\mathsf{N}(\mathtt{t}_\forall(\mathtt{t}_\exists(\varphi))) = \emptyset$ and:

$$\begin{aligned} \mathbf{T}_\exists(\varphi) &= \forall\{x^+, y^+, p_1, p_2, z^-\}\mathbf{T}_\forall(\mathtt{t}_\exists(\varphi)) \\ &= \forall\{x^+, y^+, p_1, p_2, z^-\}\exists\{z^+, p_3\}\mathbf{T}_\exists(\mathtt{t}_\forall(\mathtt{t}_\exists(\varphi))) \\ &= \forall\{x^+, y^+, p_1, p_2, z^-\}\exists\{z^+, p_3\}\mathtt{t}_\forall(\mathtt{t}_\exists(\varphi)) \\ &= \forall\{x^+, y^+, p_1, p_2, z^-\}\exists\{z^+, p_3\} \left(\begin{pmatrix} p_3 \leftrightarrow \chi[x/z^+] \\ \wedge\ \neg p_3 \rightarrow (z^+ \leftrightarrow z^-) \end{pmatrix} \wedge \left(\begin{pmatrix} p_1 \leftrightarrow (\psi[x/x^+] \wedge \neg p_3) \\ \wedge\ \neg p_1 \rightarrow (x^+ \leftrightarrow x^-) \\ \wedge\ \ p_2 \leftrightarrow \rho[y/y^+] \\ \wedge\ p_2 \rightarrow (y^+ \leftrightarrow y^-) \end{pmatrix} \rightarrow (p_1 \wedge \neg p_2) \right) \right). \end{aligned}$$

The formulas $\mathbf{T}_\exists(\varphi)$ and φ are equisatisfiable, as will follow from Theorem 9 below.

▶ **Definition 7.** *Let $\varphi \in \mathcal{L}$ be a full QBF. Let $\beta \in \mathcal{L}_{\mathsf{bool}}$ and $\{p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k\}$ be as in Fact 4.*

$$\mathbf{R}(\varphi) = \beta[p_1/Q_1X_1 \cup \mathsf{N}(\varphi_1)\mathbf{T}_{Q_1}(\varphi_1), \ldots, p_k/Q_kX_k \cup \mathsf{N}(\varphi_k)\mathbf{T}_{Q_k}(\varphi_k)].$$

▶ **Example 8.** Let us compute $\mathbf{R}(\varphi)$ step-by-step for the above $\varphi = \exists x\varphi_1 \wedge \neg\forall y\rho$ with $\varphi_1 = \psi \wedge \neg\exists x\chi$ and ψ, χ, ρ boolean. First:

$$\mathtt{t}_\exists(\varphi_1) = \begin{pmatrix} p_3 \leftrightarrow \chi[x/z^+] \\ \wedge\ \neg p_3 \rightarrow (z^+ \leftrightarrow z^-) \end{pmatrix} \rightarrow (\psi \wedge \neg p_3),$$

$$\mathsf{P}(\varphi_1) = \{z^+, p_3\},$$

$$\mathsf{N}(\varphi_1) = \{z^-\};$$

$$\mathbf{T}_\exists(\varphi_1) = \forall\{z^+, p_3\}\mathbf{T}_\forall(\mathtt{t}_\exists(\varphi_1))$$

$$= \forall\{z^+, p_3\}\left(\begin{pmatrix} p_3 \leftrightarrow \chi[x/z^+] \\ \wedge\, \neg p_3 \rightarrow (z^+ \leftrightarrow z^-) \end{pmatrix} \rightarrow (\psi \wedge \neg p_3)\right).$$

Then $\mathsf{N}(\mathtt{t}_\forall(\mathtt{t}_\exists(\varphi_1))) = \emptyset$, and

$$\begin{aligned}\mathbf{R}(\varphi) &= (\exists\{x\} \cup \mathsf{N}(\varphi_1)\mathbf{T}_\exists(\varphi_1)) \wedge (\neg\forall\{y\} \cup \mathsf{N}(\rho)\mathbf{T}_\exists(\rho)) \\ &= \left(\exists\{x, z^-\}\forall\{z^+, p_3\}\left(\begin{pmatrix} p_3 \leftrightarrow \chi[x/z^+] \\ \wedge\, \neg p_3 \rightarrow (z^+ \leftrightarrow z^-) \end{pmatrix} \rightarrow (\psi \wedge \neg p_3)\right)\right) \wedge \neg\forall y\rho.\end{aligned}$$

▶ **Theorem 9.** *Let $\varphi \in \mathcal{L}$ be a full QBF. The formulas φ and $\mathbf{T}_\exists(\varphi)$ are equisatisfiable, φ and $\mathbf{T}_\forall(\varphi)$ are equivalid, and φ and $\mathbf{R}(\varphi)$ are logically equivalent.*
Furthermore, for Q being either $\exists$ or $\forall$, we have:

1. *$\mathbf{T}_Q(\varphi)$ is a prenex formula and $\mathbf{R}(\varphi)$ is a boolean combination of prenex formulas.*
2. *$\mu(\mathbf{T}_Q(\varphi)) = \mu(\mathbf{R}(\varphi)) = \mu(\varphi)$.*
3. *The lengths of $\mathbf{T}_Q(\varphi)$ and $\mathbf{R}(\varphi)$ are linear in the length of φ.*

Proof. The first statement follows from a slightly stronger result (*): For any full QBF φ of quantifier depth $n \geq 0$, and any $Q \in \{\exists, \forall\}$, the formulas φ and $Q\mathsf{N}(\varphi)\mathbf{T}_Q(\varphi)$ are logically equivalent. We prove (*) by induction on n. For the base case n is zero and any formula φ of quantifier depth 0 is boolean, so $\mathsf{N}(\varphi) = \emptyset$ and $\mathbf{T}_Q(\varphi) = \varphi$, and the equivalence is immediate. For the induction step, assume the induction hypothesis holds for n. Let φ be a full QBF of quantifier depth $n+1$. Let $Q \in \{\exists, \forall\}$ and let Q' be its dual quantifier.

$$\begin{aligned}Q\mathsf{N}(\varphi)\mathbf{T}_Q(\varphi) &\equiv Q\mathsf{N}(\varphi)Q'\mathsf{P}(\varphi)\cup\mathsf{N}(\psi)\,\mathbf{T}_{Q'}(\psi)\,, \text{ for } \psi = \mathtt{t}_Q(\varphi) \\ &\equiv Q\mathsf{N}(\varphi)Q'\mathsf{P}(\varphi)Q'\mathsf{N}(\psi)\,\mathbf{T}_{Q'}(\psi) \\ &\equiv Q\mathsf{N}(\varphi)Q'\mathsf{P}(\varphi)\,\psi && \text{(by induction hypothesis)} \\ &\equiv Q\mathsf{N}(\varphi)Q'\mathsf{P}(\varphi)\,\mathtt{t}_Q(\varphi) && \text{(by definition of } \psi) \\ &\equiv \varphi && \text{(by Lemma 4.1)}\end{aligned}$$

Observe that the induction hypothesis applies because by Lemma 4.2 the quantifier depth of $\mathtt{t}_Q(\varphi)$ is n. From (*) it directly follows that for any full QBF φ, φ and $\mathbf{T}_\exists(\varphi)$ are equisatisfiable, φ and $\mathbf{T}_\forall(\varphi)$ are equivalid, and φ and $\mathbf{R}(\varphi)$ are logically equivalent.

The proof of the three items also proceeds by induction on quantifier depth:

1. Let us prove that $\mathbf{T}_Q(\varphi)$ is in prenex form. The base case $\mu(\varphi) = 0$ is trivial: φ is boolean. For the induction step suppose $\mu(\varphi) \geq 1$. Then $\mathbf{T}_Q(\varphi) = Q'\mathsf{P}(\varphi)\cup\mathsf{N}(\psi)\mathbf{T}_{Q'}(\psi)$, for ψ being $\mathtt{t}_Q(\varphi)$. The induction hypothesis applies because $\mu(\psi) = \mu(\mathtt{t}_Q(\varphi)) = \mu(\varphi) - 1$ thanks to Lemma 4.1; that is, $\mathbf{T}_{Q'}(\psi)$ is in prenex form. Hence $\mathbf{T}_Q(\varphi) = Q'\mathsf{P}(\varphi)\cup\mathsf{N}(\psi)\mathbf{T}_{Q'}(\psi)$ is so, too.
   The formula $\mathbf{R}(\varphi)$ is a boolean combination of formulas in prenex form because each $\mathbf{T}_\exists(\varphi_i)$ is in prenex form.
2. The base case $\mu(\varphi) = 0$ is trivial because then $\mathbf{T}_Q(\varphi) = \mathbf{R}(\varphi) = \varphi$. When $\mu(\varphi) \geq 1$ we have:

$$\begin{aligned}\mu(\mathbf{T}_Q(\varphi)) &= \mu(Q'\mathsf{P}(\varphi)\cup\mathsf{N}(\psi)\,\mathbf{T}_{Q'}(\psi))\,, \text{ for } \psi = \mathtt{t}_Q(\varphi) \\ &= 1 + \mu(\mathbf{T}_{Q'}(\psi)) \\ &= 1 + \mu(\psi) && \text{(by induction hypothesis)} \\ &= \mu(\varphi)\,.\end{aligned}$$

   The induction hypothesis applies because Lemma 4.2 tells us that $\mu(\mathtt{t}_Q(\psi)) = \mu(\psi) - 1$ when $\mu(\psi) \geq 1$.

3. We establish by induction on quantifier depth that

$$|\mathsf{N}(\varphi)| + \left\|\mathbf{T}_Q(\varphi)\right\| \leq \mu(\varphi) + 7\mathsf{NBlock}(\varphi) + 5\mathsf{NQVar}(\varphi) + \|\varphi\|.$$

The base case is immediate. For the induction step, let $\psi = \mathtt{t}_Q(\varphi)$. First of all, we observe that $|\mathsf{N}(\varphi)| = \mathsf{NQVar}(\varphi) - \mathsf{NQVar}(\psi)$ and $|\mathsf{P}(\varphi)| = \mathsf{NQVar}(\varphi) - \mathsf{NQVar}(\psi) + \mathsf{NBlock}(\varphi) - \mathsf{NBlock}(\psi)$, and thus

$$|\mathsf{N}(\varphi)| + |\mathsf{P}(\varphi)| + \mathsf{NBlock}(\psi) + 2\mathsf{NQVar}(\psi) = \mathsf{NBlock}(\varphi) + 2\mathsf{NQVar}(\varphi). \tag{**}$$

We therefore have:

$$\begin{aligned}
&|\mathsf{N}(\varphi)| + \left\|\mathbf{T}_Q(\varphi)\right\| \\
&= |\mathsf{N}(\varphi)| + 1 + |\mathsf{P}(\varphi)| + |\mathsf{N}(\psi)| + \left\|\mathbf{T}_{Q'}(\psi)\right\| \\
&\leq |\mathsf{N}(\varphi)| + 1 + |\mathsf{P}(\varphi)| + \mu(\psi) + 7\mathsf{NBlock}(\psi) + 5\mathsf{NQVar}(\psi) + \|\psi\| && \text{(by induction hypothesis)} \\
&\leq 1 + \mu(\psi) + |\mathsf{N}(\varphi)| + |\mathsf{P}(\varphi)| + \mathsf{NBlock}(\psi) + 2\mathsf{NQVar}(\psi) + 6\mathsf{NBlock}(\psi) + 3\mathsf{NQVar}(\psi) + \|\psi\| \\
& && \text{(by rearranging terms)} \\
&\leq \mu(\varphi) + |\mathsf{N}(\varphi)| + |\mathsf{P}(\varphi)| + \mathsf{NBlock}(\psi) + 2\mathsf{NQVar}(\psi) + 6\mathsf{NBlock}(\varphi) + 3\mathsf{NQVar}(\varphi) + \|\varphi\| \\
& && \text{(by Lemma 4.2)} \\
&\leq \mu(\varphi) + \mathsf{NBlock}(\varphi) + 2\mathsf{NQVar}(\varphi) + 6\mathsf{NBlock}(\varphi) + 3\mathsf{NQVar}(\varphi) + \|\varphi\| && \text{(by equality (**))} \\
&\leq \mu(\varphi) + 7\mathsf{NBlock}(\varphi) + 5\mathsf{NQVar}(\varphi) + \|\varphi\| && \text{(by rearranging terms)}
\end{aligned}$$

Keeping in mind that $\mu(\varphi) \leq \|\varphi\|$ and that $\mathsf{NBlock}(\varphi) + \mathsf{NQVar}(\varphi) \leq \|\varphi\|$, we have

$$|\mathsf{N}(\varphi)| + \left\|\mathbf{T}_Q(\varphi)\right\| \leq 9\|\varphi\|.$$

Hence the lengths of $\mathbf{T}_\exists(\varphi)$, $\mathbf{T}_\forall(\varphi)$, and $\mathbf{R}(\varphi)$ are all linear in the length of φ.
This ends the proof of the theorem. ∎

For the fragment of full QBFs of quantifier depth less or equal k, we establish the lower bound of model checking via an existing reduction of $\Sigma^{\mathsf{P}}_{k-1}$-complete problems to Θ^{P}_k problems. The latter is the class of problems that can be solved by deterministic Turing machines in polynomial time with at most logarithmic many calls to a $\Sigma^{\mathsf{P}}_{k-1}$ oracle.

▶ **Lemma 10** (Immediate from Theorem 3.2 in [10]). *Let $A = \{\varphi_1, \ldots, \varphi_n\}$ and $B = \{\psi_1, \ldots, \psi_n\}$ be sets of fully quantified QBFs in prenex form starting with an $\exists$ quantifier and of depth k such that $\varphi_i \to \varphi_{i+1}$ and $\psi_i \to \psi_{i+1}$ are valid, for $1 \leq i < n$. Deciding whether there exists $i \leq n$ such that φ_i is satisfiable and ψ_i is unsatisfiable is $\Theta^{\mathsf{P}}_{k+1}$-hard.*

▶ **Corollary 11.** *The satisfiability, validity, and model checking of full QBFs are* PSPACE*-complete problems. For the fragment of formulas of quantifier depth less or equal k, for $k \geq 0$: the satisfiability problem is $\Sigma^{\mathsf{P}}_{k+1}$-complete; the validity problem is Π^{P}_{k+1}-complete; and the model checking problem is $\Theta^{\mathsf{P}}_{k+1}$-complete.*

Proof. For each of the first three classes PSPACE, $\Sigma^{\mathsf{P}}_{k+1}$, and Π^{P}_{k+1}, the hardness result follows directly from the corresponding results for QBFs in prenex form, and the membership result follows from Theorem 9. It remains to establish $\Theta^{\mathsf{P}}_{k+1}$-completeness of model checking formulas φ of quantifier depth less or equal k in a valuation V.

The $\Theta^{\mathsf{P}}_{k+1}$-hardness result follows from Lemma 10: let $A = \{\varphi_1, \ldots, \varphi_n\}$ and $B = \{\psi_1, \ldots, \psi_n\}$ be sets of fully quantified QBFs in prenex form starting with an $\exists$ quantifier and of depth k such that

$\varphi_i \to \varphi_{i+1}$ and $\psi_i \to \psi_{i+1}$ are valid, for $1 \leq i < n$. There is an $i \leq n$ such that φ_i is satisfiable and ψ_i is unsatisfiable if and only if

$$V\left(\bigvee_{i \leq n}\left(\varphi_i \wedge \neg\psi_i\right)\right) = 1,$$

for some arbitrary valuation V. The latter is a model checking problem for formulas of quantifier depth k.

We establish the $\Theta^{\mathsf{P}}_{k+1}$-membership result by using that $\Theta^{\mathsf{P}}_{k+1}$ equals $\mathsf{P}_{\|}^{\Sigma^{\mathsf{P}}_k}$ (cf. Theorem 8.1 in [16] and the remark following it). The latter is the class of problems that can be solved by deterministic Turing machines in polynomial time with at most polynomially many non-adaptive parallel calls to a Σ^{P}_k oracle. Given a valuation V and a formula φ of quantifier depth k, we first replace the free variables of φ by either $\top$ if $p \in V$ or $\bot$ if $p \notin V$, resulting in a formula of the form

$$\varphi' = \beta[p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k]$$

for some substitution $\{p_1/Q_1X_1\varphi_1, \ldots, p_k/Q_kX_k\varphi_k\}$; we then compute

$$\mathbf{R}(\varphi') = \beta[p_1/Q_1X_1 \cup \mathsf{N}(\varphi_1)\mathbf{T}_{Q_1}(\varphi_1), \ldots, p_k/Q_kX_k \cup \mathsf{N}(\varphi_k)\mathbf{T}_{Q_k}(\varphi_k)];$$

finally we evaluate $\mathbf{R}(\varphi')$ in V, calling a Σ^{P}_k oracle for each of the subformulas $Q_iX_i \cup \mathsf{N}(\varphi_i)\mathbf{T}_{Q_i}(\varphi_i)$. ■

5 Discussion: Implementation and Evaluation

A straightforward implementation of our transformation could take the form of a "prenexing ⇒ optional preprocessing ⇒ solving" pipeline, where the preprocessing stage refers to standard techniques such as blocked-clause elimination and unit-propagation. One avenue for experimentation would then be to contrast the time needed by the different stages. However, compared to the other two stages, the task of prenexing with our method is computationally very cheap. As such, the experiments may tell the reader how good of a parser we wrote and how carefully we optimised some low-level code, but not much about prenexing itself.

A more appealing avenue for experimentation would be to compare an implementation of our prenexing approach to an alternative strategy (or to solving without prenexing). The main obstacle here is that the formulas/circuits and the tools we found are unsuitable, typically because they are restricted to prenex input. As to benchmarks, the QBFEVAL / QBFGallery collections are either prenex (QCIR 2023 and QCIR 2020) or contain only monotone operators (pre-2020 benchmarks in Boole format), in which case preexisting standard techniques apply just fine.[1] Some more benchmarks are described in [13], but it seems that they are no longer available.[2] As to software, we have checked format converters with QCIR input: Klieber's qcir-to-qdimacs.py[3]; Tentrup's qcir2aiger[4]; as well as solvers with QCIR input: Janota's cquesto[5] and qfun[6]; Slivosky's Qute[7]; Klieber's post-2010 GhostQ[8]; Tentrup's Quabs[9]; all of them assumed prenexed input. (We mention only one author for the sake of brevity.)

[1] https://qbf23.pages.sai.jku.at/gallery/
[2] http://www.info.univ-angers.fr/pub/damota/qbf is a dead link.
[3] https://www.wklieber.com/ghostq/qcir-converter.html
[4] https://github.com/ltentrup/quabs
[5] https://github.com/MikolasJanota/qesto
[6] https://github.com/MikolasJanota/qfun
[7] https://github.com/fslivovsky/qute/
[8] https://www.wklieber.com/ghostq/
[9] https://finkbeiner.groups.cispa.de/tools/quabs/index.html

Actually we seem here to have hit a chicken-and-egg problem: if preprocessors and solvers target already prenexed formats, then someone who wants to leverage QBF technology to solve their problem has to write their instance in prenex form. If instances, benchmarks, and competitions use prenex form, and if no good prenexing algorithms are known, then the incentive to develop and maintain QBF tools able to handle non-prenex input remains small. We contribute to breaking this cycle by providing an easy-to-implement prenexing algorithm with optimal asymptotic performance.

Given the scarcity of relevant benchmarks and software, the alternative would be to contrast an implementation of our proposed approach with that of a naive prenexer that we would implement ourselves on benchmarks that we also create ourselves and show experimentally that our proposed approach scales better than a naive prenexer that incurs an exponential blow-up. Such results would not increase our own confidence (let alone the confidence of an independent reader) in the merit of the proposed approach beyond what is already shown by our theoretical results, and they may even seem contrived.

Our transformation applies to formulas represented as strings, as opposed to a representation by directed acyclic graphs (DAGs). The latter allow structure sharing and thereby a more compact representation. We leave a version of our transformation for DAGs to future work.

An example is the full version of the QCIR format, which is a non-prenex, non-CNF format for QBF solvers. Full QCIR covers full QBF, but not the other way round. However, full QBF does not cover QCIR because the latter allows "structure sharing": it is not clear how a circuit/DAG (QCIR) can be converted to a formula/tree (full QBF) without incurring either an exponential blow-up in the size or an increase in the quantifier depth. We elected to focus the presentation on formulas rather than circuits in this paper because prenexing formulas seemed to be a more elementary problem than prenexing circuits. We believe that our technique can be adapted in order to convert full QCIR to prenex QCIR under the same size guarantees; the details however remain to be elaborated and are left to future work.

6 Conclusion

We have introduced a prenexing transformation for the full language of quantified boolean formulas that is polynomial and preserves quantifier depth. As far as we know we are the first to improve the known exponential reduction. This has allowed us to derive complexity results for satisfiability, validity, and model checking. In particular, we have provided a new complexity result for the polynomial hierarchy of full QBFs. Indeed, while it is immediate to adapt the textbook proofs of PSPACE membership from prenex QBFs to full QBFs, things get more subtle when looking at bounded depth formulas. Our Corollary 11 that model checking full QBFs of depth k is $\Theta^{\mathsf{P}}_{k+1}$-hard contrasts with the known D^{P}_{k} upper bound of model checking prenex QBFs. It follows that any proof of a tight upper-bound on the model checking of prenex QBFs would be too strong to be applicable to full QBFs, unless the polynomial hierarchy collapses to the k-th level (which would be a consequence of showing a $\Theta^{\mathsf{P}}_{k+1}$-hard problem to be in D^{P}_{k}) [7].

While these results are theoretical, they can be expected to have practical impact whenever applications lead to formulas with arbitrarily nested quantifiers and boolean operators.

References

1 Uwe Egly, Martina Seidl, Hans Tompits, Stefan Woltran, and Michael Zolda. Comparing different prenexing strategies for quantified boolean formulas. In Enrico Giunchiglia and Armando Tacchella, editors, *Theory and Applications of Satisfiability Testing, 6th International Conference, SAT 2003. Santa Margherita Ligure, Italy, May 5-8, 2003 Selected Revised Papers*, volume 2919 of *Lecture Notes in Computer Science*, pages 214–228. Springer, 2003. doi:10.1007/978-3-540-24605-3_17.

2 Uwe Egly, Martina Seidl, and Stefan Woltran. A solver for QBFs in nonprenex form. In Gerhard Brewka, Silvia Coradeschi, Anna Perini, and Paolo Traverso, editors, *ECAI 2006, 17th European Conference on Artificial Intelligence, August 29 - September 1, 2006, Riva del Garda, Italy, Including Prestigious Applications of Intelligent Systems (PAIS 2006), Proceedings*, volume 141 of *Frontiers in Artificial Intelligence and Applications*, pages 477–481. IOS Press, 2006.

3 Uwe Egly, Martina Seidl, and Stefan Woltran. A solver for QBFs in negation normal form. *Constraints An Int. J.*, 14(1):38–79, 2009. URL: https://doi.org/10.1007/s10601-008-9055-y, doi:10.1007/S10601-008-9055-Y.

4 Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella. Quantifier structure in search-based procedures for QBFs. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 26(3):497–507, 2007. doi:10.1109/TCAD.2006.888264.

5 Frederik Harwath. A note on the size of prenex normal forms. *Inf. Process. Lett.*, 116(7):443–446, 2016. URL: https://doi.org/10.1016/j.ipl.2016.03.005, doi:10.1016/J.IPL.2016.03.005.

6 Simone Heisinger, Maximilian Heisinger, Adrian Rebola-Pardo, and Martina Seidl. Quantifier shifting for quantified boolean formulas revisited. In Christoph Benzmüller, Marijn J. H. Heule, and Renate A. Schmidt, editors, *Automated Reasoning - 12th International Joint Conference, IJCAR 2024, Nancy, France, July 3-6, 2024, Proceedings, Part I*, volume 14739 of *Lecture Notes in Computer Science*, pages 325–343. Springer, 2024. doi:10.1007/978-3-031-63498-7_20.

7 Edith Hemaspaandra, Lane A Hemaspaandra, and Harald Hempel. A downward collapse within the polynomial hierarchy. *SIAM Journal on Computing*, 28(2):383–393, 1998.

8 William Klieber, Samir Sapra, Sicun Gao, and Edmund M. Clarke. A non-prenex, non-clausal QBF solver with game-state learning. In Ofer Strichman and Stefan Szeider, editors, *Theory and Applications of Satisfiability Testing - SAT 2010, 13th International Conference, SAT 2010, Edinburgh, UK, July 11-14, 2010. Proceedings*, volume 6175 of *Lecture Notes in Computer Science*, pages 128–142. Springer, 2010. doi:10.1007/978-3-642-14186-7_12.

9 Florian Lonsing and Armin Biere. Integrating dependency schemes in search-based QBF solvers. In Ofer Strichman and Stefan Szeider, editors, *Theory and Applications of Satisfiability Testing - SAT 2010, 13th International Conference, SAT 2010, Edinburgh, UK, July 11-14, 2010. Proceedings*, volume 6175 of *Lecture Notes in Computer Science*, pages 158–171. Springer, 2010. doi:10.1007/978-3-642-14186-7_14.

10 Thomas Lukasiewicz and Enrico Malizia. A novel characterization of the complexity class θ_k^p based on counting and comparison. *Theor. Comput. Sci.*, 694:21–33, 2017. URL: https://doi.org/10.1016/j.tcs.2017.06.023, doi:10.1016/J.TCS.2017.06.023.

11 David A. Plaisted and Steven Greenbaum. A structure-preserving clause form translation. *J. Symb. Comput.*, 2(3):293–304, 1986. doi:10.1016/S0747-7171(86)80028-1.

12 Igor Stéphan. Functional semantics for non-prenex QBF. In Béatrice Duval, H. Jaap van den Herik, Stéphane Loiseau, and Joaquim Filipe, editors, *ICAART 2014 - Proceedings of the 6th International Conference on Agents and Artificial Intelligence, Volume 1, ESEO, Angers, Loire Valley, France, 6-8 March, 2014*, pages 358–365. SciTePress, 2014. doi:10.5220/0004760303580365.

13 Igor Stéphan and Benoit Da Mota. A unified framework for certificate and compilation for QBF. In Ramaswamy Ramanujam and Sundar Sarukkai, editors, *Logic and Its Applications, Third Indian Conference, ICLA 2009, Chennai, India, January 7-11, 2009. Proceedings*, volume 5378 of *Lecture Notes in Computer Science*, pages 210–223. Springer, 2009. doi:10.1007/978-3-540-92701-3_15.

14 Leander Tentrup. Non-prenex QBF solving using abstraction. In Nadia Creignou and Daniel Le Berre, editors, *Theory and Applications of Satisfiability Testing - SAT 2016 - 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings*, volume 9710 of *Lecture Notes in Computer Science*, pages 393–401. Springer, 2016. doi:10.1007/978-3-319-40970-2_24.

15 Gregory S. Tseytin. On the complexity of derivation in propositional calculus. In A.O. Slisenko, editor, *Studies in Constructive Mathematics and Mathematical Logic, Part II, Seminars in Mathematics*, pages 115–125. Steklov Mathematical Institute, 1970. Translated from Russian: Zapiski Nauchnykh Seminarov LOMI 8 (1968), pp. 234–259.

16 Klaus W. Wagner. Bounded query classes. *SIAM J. Comput.*, 19(5):833–846, 1990. doi:10.1137/0219058.